

Jean Guy Meunier.

Représentation et non computationnalité.

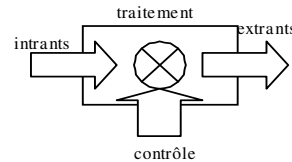
traduction de l'italien
in *Representazione e teoria della mente*. Padovani, G & Braga I. P. Univ Parma. 2002
p 211-229.

1-Les projets de naturalisation fonctionnelle.

Les projets de naturalisation de l'esprit abondent. Ils nous proposent une compréhension causale de ce phénomène complexe que sont la perception, le désir, la conscience, etc. Ils refusent l'explication sémiotique, phénoménologique, herméneutique. On préfère une lecture soit fonctionnaliste à la Fodor, soit réductionniste à la Churchland-Changéux ou encore dynamique à la Petitot ou à la Van Gelder. Certains, comme Dennett, Cummins, Dretske tentent un compromis entre un respect du sens, de l'intentionnalité et de la causalité. Malgré des différences importantes dans les concepts et les arguments, on peut voir que l'ensemble de ces projets de naturalisation de l'esprit s'inspirent souvent d'un même univers de modèles, qui, malheureusement, ne sont pas toujours apparents dans l'explication. De fait, ils opèrent presque tous dans des variantes plus ou moins transparentes de la théorie des systèmes complexes de traitement de l'information (STCI), dont une des plus connues, le *fonctionnalisme*, postule que les STCI sont de types représentationnels, symboliques et surtout computationnels.

Dans ce modèle, une procédure fonctionnelle est définie comme une machine abstraite qui reçoit des données

d'entrée, les traite, les contrôle et en produit des intrants. (voir schéma suivant)



Bien que simple, ce schéma classique permet de voir que la "machine cognitive" repose avant tout sur la présence d'un mécanisme de contrôle. Si celui-ci prend la forme de règles, d'algorithmes ou de fonctions récursives alors le traitement sera dit *computationnel*. Autrement dit, dans ce modèle, traiter de l'information c'est la *computer*.

2-La thèse Church Turing

Tout cet édifice du fonctionnalisme repose évidemment sur ce concept de *computationnalité* c'est-à-dire de "calculabilité"¹. Malheureusement, malgré ce qu'on peut en penser, ce concept n'est pas aussi limpide qu'on le croit. De fait, il n'existe pas un consensus clair sur la nature de ce qu'est la computationnalité ou du moins ses constituants essentiels. De plus, lorsqu'appliquée à une théorie cognitive, elle n'est pas sans problème. Pour notre propos, rappelons ici quelques éléments de cette problématique. Pour qu'un traitement fonctionnel puisse être computationnel, il doit répondre à des conditions strictes. En effet, au sens technique de ce terme, une procédure ou un ensemble de procédures est dite *computationnelle* si elle contient des opérations *effectives* (n.b. non efficaces) et *fiabiles* pour reconnaître ou produire les éléments ou des configurations d'éléments

dans le "calcul "d'une fonction. "Effectif" signifie que ces opérations avec un nombre fini d'instructions entraîne toujours un résultat assuré. Un exemple classique d'une procédure computationnelle est la table de vérité du langage des propositions. Il est en effet toujours possible de déterminer d'une formule arbitraire de ce langage via cette table la vérité.

Ce concept de computationnalité a de plus reçu plusieurs traductions. Church et Turing, et Post² ont démontré par divers arguments que, malgré des apparentes différences, une fonction computationnelle était équivalente à une *fonction récursive*, un *algorithme*³, une *grammaire* et finalement une *machine de Turing*. Ceci constitue l'essentiel de la thèse dite de Turing Church :

« La même classe de fonctions partielles (et donc de fonctions totales) est obtenue dans chaque cas " (Rogers 1987:19)

A son origine, cette thèse, sous ses multiples formes était surtout appliquée à l'univers mathématique et plus spécifiquement encore à celui des nombres. Devant sa puissance cependant, on proposa évidemment de l'étendre à d'autres domaines en l'occurrence le traitement informatique c'est-à-dire l'ordinateur. On dira alors que si une fonction est computationnelle et donc une machine de Turing, elle est réalisable matériellement dans un ordinateur. L'ordinateur n'étant cependant qu'une forme physique parmi d'autres, on étendra la thèse aux machines biologiques tel le cerveau. Il n'en fallait pas plus pour faire un autre petit saut: En effet, si le cerveau est une machine computationnelle, la pensée n'en serait-elle pas une machine aussi? Penser n'est-il pas computer comme le disait Hobbes ou encore comme le formulait avec emphase : Rumelhart et McLelland" : "*Minds are what*

brain do best!". Thèse qui dans le discours vulgarisateur se traduit ainsi:

Put the right kind of soft ware in a computer, and it will do whatever you want it to. There may be limits on what you can do with the machines themselves, but there are no limits on what you can do with software"(1984 avril editorial: Times)

C'est par ce transfert que certains chercheurs ont avancé que la cognition était aussi de nature computationnelle. Thèse qu l'on retrouve sous une forme ou autre dans les théories formelles du comportement intelligents (Turing, 1936; Ashby, 1956; Von Neumann, 1958 ; Pylyshyn, 1984, etc.), de l'intelligence artificielle (Newell et Simon 1976), de la philosophie logico-analytique (Fodor, 1975 ; Haugeland, 1986), de la psychologie cognitive (Johnson-Laird, 1988) de la linguistique (Jakendoff, 1988 ; Pinker, 1998), etc.

3- Les limites de la thèse Church Turing.

La thèse de Church Turing fut évidemment une contribution importante pour le développement des technologies informatiques. Mais pour l'appliquer aux structures cognitives, elle a dû être l'interprétée dans deux sens tout à fait particuliers que l'on peut résumer ainsi:

- a- tout traitement informatique réalise une fonction computationnelle,
- b- toute fonction computationnelle est réalisable dans un ordinateur physique.

Or ces deux lectures sont problématiques et sont à la source de plusieurs confusions philosophiques. En effet, ces lectures semblent affirmer qu'il y a équivalence entre une machine informatique et une fonction computationnelle c'est-à-dire une machine de Turing. Or ce n'est pas ce que la thèse dit.

En effet la thèse Church Turing établit uniquement une équivalence formelle entre des fonctions computationnelles, les fonctions récursives, les algorithmes et les machines de Turing. Elle n'affirme aucunement l'équivalence entre cette classe de fonctions et un ordinateur.

«This loosening of established terminology is unfortunate, for neither Church nor Turing endorsed, or even formulated, this further proposition. [i.e. that].... Thesis M: Whatever can be calculated by a machine (working on finite data in accordance with a finite program of instructions) is Turing-machine-computable. (Copeland : 1996)

Ces deux lectures sont donc problématiques. La première, à savoir : *toute fonction computationnelle est réalisable dans un ordinateur physique*, n'est aucunement vraie. En effet, la recherche et surtout l'expérience a montré que, pour être réalisée, certaines fonctions computationnelles complexes (NP complexe) pouvaient exiger des ressources et du temps infini et donc rendre en pratique le traitement informatique impossible. Un ordinateur peut en effet être limité dans sa vitesse de traitement, sa mémoire, son archivage, l'ampleur de ses données et donc empêcher la réalisation d'une fonction complexe. C'est le problème de la tractabilité des algorithmiques. (Harel,1987; Arbid, 1987,1995). Bref, ce n'est pas parce qu'une fonction est calculable qu'elle est automatiquement réalisable par un ordinateur.

La deuxième lecture, à savoir: *tout traitement informatique réalise une fonction computationnelle*, est aussi problématique. Elle donne l'impression que le traitement effectuée par un ordinateur réalise automatiquement une fonction calculable. Or tel n'est pas le cas. La technologie informatique présente en effet de nombreux programmes qui opèrent sur des problèmes

que l'on sait formellement être non calculables. De fait, on sait, par exemple que les fonctions suivantes sont non calculables:

- a) en mathématiques: la conjoncture de Golbach ⁴ et le Entscheidungs problem (Turing,1936)
- b) en logique: la preuve de la vérité des formules du langage des prédicats du premier ordre.
- c) en informatique: l'arrêt des algorithmes (halting problem Turing: 1936), la vérification des programmes
- d) en linguistique: le Word group problem (Novikov, (1955⁵), la reconnaissance de phrases dans l'intersection de langage de type 1. (Gross et Lantin 1970; Partee et al., 1993)
- e) en topologie: le recouvrement de surface par des tuiles de formes géométriques complexes (Penrose 1994).

De fait, il existe un nombre 2^{\aleph} c'est-à-dire continu et indénombrable de fonctions non calculables c'est-à-dire de fonctions pour lesquelles il n'existe pas de solutions algorithmiques (Rogers: 1967: 22)

Ces deux lectures nous mettent cependant devant des paradoxes technologiques. En effet , dans un premier cas, on sait qu'il existe certains programmes qui traitent des fonctions que l'on sait formellement être *NP complexes* et non tractables. Pensons au calcul de certaines d'intégrales. Comment un ordinateur peut il réaliser une fonction qui, bien que computationnelle, est si complexe qu'on sait qu'il est impossible de la réaliser dans un temps utile et pratique? De même, un deuxième cas, on sait qu' il existe des programmes qui réalisent des opérations reliées à des fonctions reconnues a priori non-computables. N'existe-t-il pas démonstrateurs de théorèmes c'est-à-dire des

programmes qui, pour une formule du langage du premier ordre, en vérifient la vérité? N'existe-t-il pas aussi des vérificateurs de programmes? Des parseurs pour des langages formellement non calculables?

Ainsi, si toute fonction NP complexe n'est pas concrètement réalisable et si toute fonction non calculable n'est pas computable, quel genre d'opérations réalisent un ordinateur lorsqu'il traite de tels cas? La réponse à cette question est relativement simple et classique. Pour rendre ces fonctions NP complexes ou non calculables traitables par un ordinateur, il s'agit de faire du "soft computing" (Van Gelder, 1996) c'est-à-dire de délester certaines contraintes sur les conditions de la stricte computationnalité. Ceci se réalisera de deux manières différentes. Une première purement pragmatique et une seconde, sémantique. Dans la première forme de *soft computing*, on trouvera des solutions pratiques efficaces. Par exemple, on acceptera que dans le cas des problèmes NP complexes on y substitue des algorithmes qui soient tractables et diminuent la complexité. Dans le cas de la non calculabilité, on acceptera de prendre des solutions ad hoc pour certains problèmes spécifiques. Ainsi ces fonctions deviendront traitables informatiquement via des *approximations*, des *heuristiques* ou des *oracles*. La solution de approximations bien que formellement non satisfaisante donne des solutions acceptables pour de nombreux problèmes pratiques. Pour leur part les heuristiques impliquent des interventions ad hoc des "rules of thumb" qui permettent de produire des solutions peut être pas effectives mais tout au moins efficace.⁶ Enfin, les oracles, permettent sous certaines conditions des procédures de transformations de fonctions en fonctions décidables et donc computables.

La deuxième solution est sémantique c'est-à-dire, elle modifie la définition même de la computationnalité. Ainsi elle abandonnera certaines contraintes fortes telles l'atomicité des informations c'est-à-dire leur non continuité, la systématique compositionnelle et la séquentialité. Dans cette perspective, on considérera quand même computationnelle une procédure qui n'opère plus sur des symboles, qui n'est plus compositionnelle et séquentielle et qui sera souvent parallèle et probabiliste. Et on placera dans le club du "*soft computing*" les modèles connexionnistes (Rumelhart et McClelland, 1987), morpho dynamiques (Thom, 1975; Petitot, 1985; Franklin, 1995) génétiques (Holland, 1975) émergentistes (Maturana et Varela, 1980, Forest 1991) dynamiques (Van Gelder, 1997) évolutifs (Dawkins 1982, Scott Kelso, 1995) quantiques (Penrose 1994). Dans cette solution, on parle encore de "computation" mais dans un sens élargi qui ne correspond plus au sens original stricte de la computationnalité.

On voit ainsi que la compréhension de la computation peut être étendue. Dans une théorie stricte de computation établit des relations d'équivalence entre des classes de fonctions. Mais elle n'établit aucunement une équivalence automatique entre des machines abstraites et des machines concrètes que celles-ci soient un ordinateur ou un cerveau. Dans cette compréhension stricte, un ordinateur ne compute que des fonctions calculables. Il est vraiment la matérialisation d'une machine de Turing.

Dans une conception étendue, on peut rompre avec cette dernière définition et accepter comme computationnelles les opérations que réalisent certaines machines physiques bien que ces opérations ne soient pas computationnelles au sens stricte de ce terme. Autrement dit ces machines ne réaliseront pas une computation forte. On parlera alors de computation molle!

4- Computationnalité dure et computationnalité molle.

Qu'en est-il alors du cerveau et de la pensée? Leur opérations réalisent t-elle une computationnalité forte ou une computationnalité molle?

Le fonctionnaliste classique défend la thèse de la computationnalité dure. Pour un Fodor et un Pylyshyn (1986), le cerveau, en tant qu'outil du mental (mind) doit traiter des fonctions symboliques, compositionnelles et interprétables. Au sens strict, le cerveau traite des fonctions calculables. Et donc les comportements issus de l'activité cérébrale (le behavior) sont issus de cette computationnalité.

Pourtant, même Turing n'aurait pas défendu une telle thèse. Il faut rappeler ici qu'à l'origine la thèse de Turing visait la description de certains comportements humains très spécifiques c'est-à-dire, ceux qui sont répétitifs, comme certains gestes automatiques, ritualisés, etc.. Et ce sont strictement ces comportements répétitifs que Turing voyait comme pouvant être simulés par une machine de Turing:

«We may compare a man in the process of computing a... number to a machine. (Turing 1936: 231.) Nous soulignons.

Autrement dit, pour Turing, il y a des comportements qui sont de type algorithmiques et que l'on peut traduire en machine de Turing. Ils sont peut être aussi très nombreux, mais il ne s'ensuit pas que tous les comportements sont de ce type. Qui plus est, il est clair pour lui que seulement un sous ensemble particulier de problèmes résolus par les humains peuvent être réalisés par des machines et même simulés par ordinateur (voir le test de Turing).

The class of problems capable of solution by the machine can be defined fairly specifically. They are [a subset of] those problems which can be solved by human clerical labour, working to fixed rules, and without understanding. (Turing 1946: 38-9.)

Autrement dit, la thèse philosophique de Turing Church ne dit pas que tous les comportements humains sont simulables par une machine computationnelle. Au contraire, il n'y en qu'une sous-classe particulière, à savoir celle qui présente une certaine répétitivité. Ce sont uniquement ces sous classes qui sont Turing réalisables.

Si tel est le cas, ceci nous amène à revoir en profondeur la thèse computationnelle de l'esprit. En effet, il faut s'interroger sur la nature des opérations qu'effectue un cerveau. Réalise-t-il toujours des opérations qui sont formellement computables? Formellement NP complexe? Que fait le cerveau dans ce cas? En effet, comment pourrions-nous être assurés qu'il réalise de telles fonctions alors qu'un ordinateur, lui, ne peut même pas les réaliser. Autrement dit, qu'est-ce qui nous assure que toutes opérations qu'un cerveau peut réaliser sont des fonctions NP complexes et computables? Ce pourrait-il qu'un cerveau comme un ordinateur doive traiter des fonctions non calculables? NP complexes? Ne réalise-t-il pas lui aussi une computation molle? Si oui, alors ne pourrait-il pas lui aussi en appeler à des heuristiques, des "oracles" ou des approximations pour apporter des solutions concrètes pratiques et efficaces à des problèmes NP complexes ou non calculables? Mais si tel est le cas, d'où viendraient ces heuristiques? Ces oracles? qui balisera les approximations?

L'avantage de l'approche computationnelle dure est qu'elle permet d'emballer les fonctions générales de manière close et compactes et donc d'offrir des programmes

bien ficelés qui sont alors transmissibles comme des " packages". On peut les mettre dans des disquettes ou des gènes. Mais dans une computation molle, les heuristiques, les approximations, les oracles viennent briser ce modèle. Ils exigent des interventions d'un tiers pour baliser, transformer ou affecter le programme lui-même. Pour donner ces instructions, il faut donc un nouveau type d'homuncule, il faut un " ingénieur mental". Ce qui n'est pas sans problème. Car si le cerveau fonctionne avec de telles heuristiques, approximations et oracles, il faudra lui fournir aussi un ingénieur virtuel.

Ceci est possible selon les théoriciens de la "soft computing" . Car ce qui caractérise ce type de traitement d'information est qu'il est réglé ou contrôlé par quelque chose d'extérieur au système lui même, à savoir les *paramètres*. (Scott Kelso, Varela, Petitot etc). Ce sont eux qui contrôlent les degrés de liberté et leur pôles de convergences: les attracteurs. Mais tout comme dans le cas précédent, il faut se demander d'où viennent ces paramètres? Sont-ils véritablement externes au système ou lui sont-ils encore internes et dans ce cas relèvent-ils d'un autre niveau ou d'un autre type de contrôle? Sont-ils transmis? Émergent-ils de l'environnement. Sont-ils appris ? etc.

5- La représentationnalité.

Revenons maintenant à la question de la représentation. Ces systèmes de traitement complexes d'information sont-ils encore des systèmes qui manipulent des représentations. De fait, comme on le sait, le concept de computationnalité en sciences cognitives, est intimement lié à celui de représentation. Et en conséquence, il formerait un tout avec d'autres: tel par exemple la séquentialité, la cyclicité, etc.

*These property –représentation,
computation, sequential and cyclic opération,*

*and homuncularity, form a mutually
interdependent cluster." Van Gelder 1997:
426)*

En conséquence si, selon plusieurs chercheurs, en raison de la complexité et de la non calculabilité de plusieurs processus cérébraux et mentaux, il faut préférer une computationnalité molle, ne doit-on pas aussi abandonner une théorie de la représentation? Répondre à cette question n'est pas simple. Tout repose sur la compréhension que nous avons du concept de *représentation* qui est en jeu, qui on le sait peut être compris dans plusieurs sens.

Le plus classique et le plus contesté, du moins dans les sciences cognitives, le rend synonyme de représentation *symbolique*. On dira ainsi qu'une est "représentation", est symbolique parce qu'elle est à propos de quelque chose d'autre.

"Nous traitons avec des machines à traiter de l'information. Dans notre terminologie, ceci signifie que le mode de fonctionnement de ces machines consiste à utiliser des symboles qui tiennent la place des choses qu'ils représentent. " (Marr 1982 : 21).

Et c'est à cette conception que plusieurs chercheurs s'attaquent (Winograd et Flores 1986 ; Moravec,1988 ; Varela, 1988 ; Chalmers 1991 ; Van Gelder, 1997 ; Petitot 1985; Scott-Kelso 1995; Brooks, 1997 ; Franklin, 1995 ; etc.). On la juge totalement inadéquate pour la compréhension des systèmes complexes du traitement de l'information, surtout ceux de types " intelligents " :

*" It is the concept of representation
which is insufficiently sophisticated. " (Van
Gelder 1993 : 6).*

*" Representation is the wrong unit of
abstraction in building the bulkiest parts of
intelligent systems. " (Brooks, R. A. 1997:
396).*

“ *We are not building representations at all!* ” (Thelen et Smith 1994 : 338).

Pour notre part, nous croyons, comme nous l'avons défendu ailleurs, (Meunier, 2000) que le concept de représentation tel que défini tant par les tenants de la computation forte que de la computation molle, ne correspond que peu au concept classique de la représentation. En effet dans presque toute la tradition philosophique, le concept de représentation ne signifie pas "symbole". Dans son sens le plus classique en effet, il signifie simplement "*re-présence*"⁷ d'un monde extérieur dans un système c'est-à-dire présence originale du monde externe dans le monde interne du sujet ou en terme moderne trace ou inscription de l'environnement dans le système. Certes, cette re-présence ou cette inscription peut prendre la forme de symbole. Mais plusieurs autres formes sont aussi possibles. Pour Peirce, par exemple, la forme symbolique n'est qu'une forme parmi d'autres de "représentation". Pour lui, une représentation pourrait être aussi bien iconique que indicielle, c'est-à-dire causale. De plus rien n'empêche qu'elle pourrait être dispositionnelle, non causale, non linéaire, distribuée, paramétrisée, etc.

Autrement dit, dans son sens classique, le concept de représentation demeure une façon commode de parler de certains systèmes en tant qu'ils possèdent en eux une certaine présence de leur environnement et qu'ils en gardent soit la *trace*, *l'inscription*, *la sensation*, *la capture*, *l'engramme*, *la perception*, etc. Mais quelque soit l'expression ou le cadre théorique que l'on privilégie, on devra toujours parler d'un mode de re-présence de l'environnement externe dans l'agent Et on devra s'engager sur au moins trois thèses a) une thèse sémiotique: Ce qui est traité par un système est relié d'une manière ou d'une autre à ce qui lui est extérieur. b) une thèse

épistémique: Un système effectue sur ses objets des transformations qui sont originales par rapport à celles qu'on trouve habituellement dans le monde physique c) une thèse ontologique : le traitement est réel et efficace. Évidemment, on ne s'entendra pas sur la nature de la relation sémiotique : est-elle symbolique, iconique ou causale? De même on divergera sur la nature du type de traitement effectuée: Est-elle purement logique, sémantique, nomologique, computationnelle, etc. ? Et que dire du statut d'existence de cet objet? Sont-ils physiques ou mentaux? Dans cette perspective, le concept de représentation compris dans son sens classique, n'est pas prêt de disparaître. Il demeure trop commode. Il permet de parler des systèmes de traitement d'information sans engager le chercheur sur les modalités de chacune de ces thèses.

À mon avis, le problème du concept de représentation dans les théories du traitement de l'information ne réside pas dans la thèse sémiotique, logique et ontologique, mais dans la trop grande généralité de ce concept. En effet, bien que nous puissions montrer que les divers modèles de représentation et d'information partagent des similarités, il n'est resté pas moins que les modèles sont incomplets.

Cette incomplétude se manifeste premièrement lorsqu'il faut préciser la nature du processus représentationnel. Dans une modélisation de type représentationnel toute la dynamique semble se limiter à distinguer ce qui se passe entre l'extérieur et l'intérieur du système. Mais cela ne répond pas à la question: Comment, quelque chose d'extérieur procède-t-elle pour "affecter" un système ou à l'inverse comment un système procède-t-il pour inscrire en lui quelque chose d'extérieur. Autrement dit, le concept de représentation ne se compromet pas sur la question cruciale de la nature du rapport causal qu'est l'inscription de l'information

dans un agent ou un système. Où commence et finit l'intérieur et l'extérieur d'un système? Deuxièmement, il ne livre pas d'explication sur les comportements complexes. En effet, la conceptualité représentationnelle donne à penser que le traitement effectué est simple et que tout "fonctionne" dans ces systèmes à un seul niveau ou avec un seul type de représentation. Autrement dit, tout comme un miroir ramène ce qu'il reflète à deux dimensions, le concept de représentation semble réduire le traitement qu'il effectue à un seul niveau. Il livre ainsi une impression d'aplatissement d'une réalité systémique complexe.

Or, la majorité des STI ne sont pas des systèmes présentant un seul niveau de traitement. Ce que montre clairement les multiples modèles de traitement de l'information. C'est pourquoi, une théorie de la représentation doit défendre une multiplicité de niveaux (Braga, 1997). Par exemple, la robotique récente de type symbolique (Albus et Meystel, 1996) ou dynamique (Brooks, 1989) ne peut fonctionner avec une conception qui ne traite qu'un seul niveau de "re-présence". Un seul niveau est insuffisant pour modéliser de tels systèmes complexes. Un robot doit non seulement interagir avec le monde, mais aussi agir lui-même sur ces paramètres eux-mêmes. Pour certains, ces niveaux seront hiérarchiques. (Pollock, 1989) pour d'autres (Clark, 1997) elles seront associatives ou agrégatives (Brooks, 1989, 1997).

On peut imaginer ce qu'il en est lorsque veut expliquer par le concept de représentation le processus aussi complexe que celui du traitement cognitif humain, dont on dit qu'ils possèdent de l'esprit! Comme l'ont montré les débats en philosophie de l'esprit, il est difficile de concevoir le comportement tant cognitif comme n'opérant qu'à un seul niveau.

Malgré cette limite, le concept de représentation demeure utile. Il est en effet, très difficile d'abandonner dans une théorie de l'information, l'architecture conceptuelle que sous-tend la notion de représentation. Cette architecture se retrouve dans les multiples théories de l'information. Ce qui n'est pas surprenant car, historiquement et, surtout, conceptuellement, *information* et *représentation* partagent de grandes similarités. En aucune manière, on ne peut identifier exclusivement la représentation et l'information à un mode symbolique ou à un mode non symbolique. Et là commencent les problèmes intéressants.

" Expliquer comment un état d'un système est capable de représenter n'est que la moitié de l'histoire; nous devons expliquer en quoi consiste, pour un état représentationnel, le fait d'être une représentation pour le système qui l'a "

Cummins (1983:93)

Ainsi, dans cette perspective, même un système non computationnel de type dynamique, connexionniste, génétique, etc. opérerait avec de la représentation. Mais cette représentation ne serait pas nécessairement symbolique, iconique ou "mentale".

S'il en était ainsi, ceci nous permettrait de continuer à croire que *penser, être conscient, désirer, se rappeler, croire*, etc, sont des activités d'un type différent de celle que le réalise la vessie, l'estomac et l'intestin. Bref, peut être le cœur physique est-il un simple pompe. Mais il nous sera toujours difficile de penser que le "cœur" psychologique est aussi une pompe. Peut être est-il au moins un machine computationnelle molle qui se nourrit de représentations?

Jean Guy Meunier

Université du Québec à Montréal

Références:

- Albus, J.S., Meystel. A. M. (1996). « A Model Architecture for Design and Implementation of Intelligent Control in Large and Complex Systems International ». *Journal of Intelligent Control and Systems*, Vol. 1, No 1 1996, pp 15-30
- Arbib, M.A., (1987). *Brains, Machines and Mathematics*. Second edition. New York: Springer.
- Arbib, M.A., Ed. (1995). *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA: MIT Press. (See pp. 4–11.)
- Ashby, R., (1956). *An introduction to Cybernetics*. New York. John Wiley.
- Braga Illa, F. (ed, (1997). *Livelli di rappresentazione*. Urbino Quattro venti.
- Brooks, R.A., (1997). Intelligence without representation. *Artificial Intelligence* 47: 139-159. In J. Haugeland, Ed., *Mind Design II: Philosophy, Psychology, Artificial Intelligence*. 2nd ed., revised and enlarged. Cambridge, MA: MIT Press, pp. 395–420.
- Brooks, R.A., (1989) A Robot that Walks: Emergent Behaviors from a Carefully evolved Network *Neural Computation*. 1: 253-62
- Churchland. P. M., (1989) *The Neurocomputational Perspectives*, Cambridge. MIT Press
- Clark, A., (1989) *Associative Engines*, Cambridge, MIT Press.
- Cummins, R., (1989) *Meaning and Mental Representation*, Cambridge, MIT Press.
- Curry, H.B. 1932. 'Some Additions to the Theory of Combinators'. *American Journal of Mathematics*, 54, 551-558.
- Davis M. (: 1965) *The undecidable*: New York
- Dawkins, R, (1982) *The Selfish Gene*, Oxford University Press
- Johnson-Laird, P. N. 1988 *The Computer and the Mind*. Cambridge: Harvard University Press.
- Dennett, D. (1978). *Brains Storms*. Cambridge: MIT Press.
- Fodor, J. A., (1975), *The Language of Thought*, New York: Crowell.
- Fodor, J.A. and Pylyshyn, Z.W., (1988) Connectionism and Cognitive Architecture: A Critical Analysis. *Cognition*, 28(1-2), 3-71.
- Forest (19XX)
- Harel, D., (1992) *Algorithmics: The Spirit of Computing*. Reading, Mass.: Addison-Wesley.
- Franklin, S., (1995) *Artificial Minds*, Mit Press.
- Gibson, J. J., (1950). *The Perception of the Visual World*. Boston: Houghton Mifflin.
- Gödel, K., (1934) On Undecidable Propositions of Formal Mathematical Systems'. Lecture notes taken by Kleene and Rosser at the Institute for Advanced Study. Reprinted in Davis, M. (ed.) 1965. *The Undecidable*. New York: Raven.
- Gross, M. Lentin.A., (1970) *Notions de Grammaires Formelles*, Paris, Gauthiers Villars.
- Holland, J.H., (1975). *Adaptation in Natural and Artificial Systems*. Ann Arbor. University of Michigan Press.
- Johnson-Laird, P. N. (1988), *The Computer and the Mind*. Harvard University Press.
- Kleene, S.C., (1936). 'Lambda-Definability and Recursiveness'. *Duke Mathematical Journal*, 2, 340-353.
- Moravec, H., (1988) *Mind Children*, Cambridge: Mass: Harvard University Press.
- Jackendoff, R., (1987). *Consciousness and the Computational Mind*. Bradford Book MIT Press.
- Markov, A.A., (1960 'The Theory of Algorithms'. American Mathematical Society Translations, series 2, 15, 1-14.
- Marr, D., (1982) *Vision: A computational Investigation into the Human Representation and Processing of Visual Information*. San Francisco: W. H. Freeman.
- Maturana, H.R et Varela, F.J., (1980) *Autopoiesis and Cognition` : The Realization of the Living*. Dordrecht, Boston, London, D Reidel.
- Meunier, J.G., (2000) Niveau de représentation et information in F" Braga " La représentation " U.de Trento: sous presses.
- Newell, A. and Simon, H., (1976) " Symbol Manipulation" in *Encyclopa of Computer Science*, New York: Petrocelli/Charter 1976.
- Novikov (1955)
- Partee, B.H, Meulen A., Wall, R.E., (1993) *Mathematical Methods in Linguistics* Kluwer Academic Publishers.
- Peirce, C. S., (1931-58) *Collected Papers*, Cambridge: Harvard University Press.)
- Penrose, R (1994) *Shadows of the mind*. Oxford U. P.
- Petitot, J. (1985). *Morphogénèse du Sens*. Presses Universitaires de France.
- Pollock, J. L. (1989). *How to Build a Person: A Prolegomenon*. Cambridge, Massachusetts: MIT Press.

- Post, E.L., (1936.) 'Finite Combinatory Processes – Formulation. *Journal of Symbolic Logic*, 1, 103-5.
- Pylyshyn, Z. (1984). *Cognition and computation*. Cambridge: MIT Press.
- Rogers H., (1967), *Theory of Recursive functions and Effective Computability*, MIT Press, edition 1987.
- Rumelhart, D. and Mc Lelland, J. L., (1987)*Parrallel Distributed Processing*, Vols. I et II. Cambridge: MIT Press.
- Schönfinkel, M., (1924). 'Über die Bausteine der mathematischen'. *Mathematische Annalen*, 92, 305-316.
- Scott Kelso, J.A., (1995) *Dynamic Patterns*, MIT Press.
- Thom, R., (1975). *Modèle mathématique de la morphogénèse*. Paris, Christian Bourgeois.
- Turing, A., "On Computable Numbers, With an Application to the Entscheidungsproblem", *Proceedings of the London Mathematical Society*, Series 2, Volume 42, 1936; reprinted in M. David (ed.), *The Undecidable*, Hewlett, NY: Raven Press, 1965
- Turing.A., (1936) "On computable numbers with an application to the Entschcheidung problem. Proc London Math Soc. 42 (1936) pp 230-265
- Varela F. J. (1988), *Invitation aux sciences cognitives*. Paris, Seuil.
- Von Gelder, (1997) T. Dynamics of Cognition. In Haugeland, (ed). J(1997) *Mind design II*, MIT Press.
- Eric Dietrich 1999: MIT ENCY algorithm p 12
- Nicolis, J. S. (1985). *Chaotic dynamics of information processing with relevance to cognitive brain functions*. 167-172.

Notes

¹ La calculabilité est l'expression technique française pour la "computationality". Cette expression cependant est ambiguë car elle semble limitée le problème des procédures décidables, effectives et finies, au seul domaine des nombres. Le terme anglais de "computation" implique une définition centrée sur les procédures et non sur le domaine auquel s'appliquent les procédures. Des fonctions sont "computables" même si elles ne s'appliquent pas au domaine des nombres.

² On a ainsi démontré que toutes les fonctions effectivement calculables investiguées «à savoir : les fonctions lambda-définissables et récursives, les formes canoniques normales de Post (1943, 1946), la logique combinatoire de Schönfinkel (1924,) Curry 1932), les algorithmes de Markov (1960), et la notion de reconnaissance de Gödel's (1936), les langages de Kleene (1952). Sont toutes interdéfinissables et traduisibles plus spécifiquement elles peuvent toutes être caractérisée par une machine de Turing.

³ Il faut distinguer entre un algorithme et une fonction calculable par un algorithme. Par ailleurs, une même fonction peut être exprimée par plusieurs algorithmes. Par exemple, la fonction du plus petit commun dénominateur de deux nombres. est une fonction calculable mais il existe plusieurs algorithmes différents pour la réaliser. Le plus célèbre est celui d'Euclide
En un sens, l'algorithme est une sorte de définition intensionnelle de la fonction calculable en ce qu'elle donne des moyens cognitivement interprétable et utilisable par un agent. (machine ou humain)
"an algorithm is an intensional definition of a special kind of function namely a computable function. The intensional definition contrasts with the extensional definition of a computable function, which is just the set of the function's inputs and outputs". —Eric Dietrich 1999: MIT ENCY algorithm p 12.

⁴ L'exemple de la conjecture de Golbach est plus connu, à savoir que tout nombre pair plus grand que 2 est la somme de deux nombres premiers. Exemple: $8=5+3$
Or, il n'existe aucun algorithme connu pour cette fonction. Il se pourrait qu'il n'en existe aucun.

⁵ il n'existe pas de procédure fonctionnelle calculable pour la transformation de certaines chaînes de lettres en autres chaînes de lettres.

⁶ [A procedure that]may solve a given problem, but offers no guarantees of doing so, is called a heuristic for that problem Newell, Shaw, Simon 1957 : 114)

⁷ Ce re-présence peut être aussi comprise au sens husserlien phénoménologique c'est-à-dire sous la modalité de la présence à la conscience. Mais la représentation prend un sens plus dynamique dans ce cas.